ETH Zürich
Professur für CAAD
Prof. Dr. Ludger Hovestadt

nds2004:caad:hbt:arch:ethz

# xCube

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

# xCube

nds2004 : caad : hbt : arch : ethz

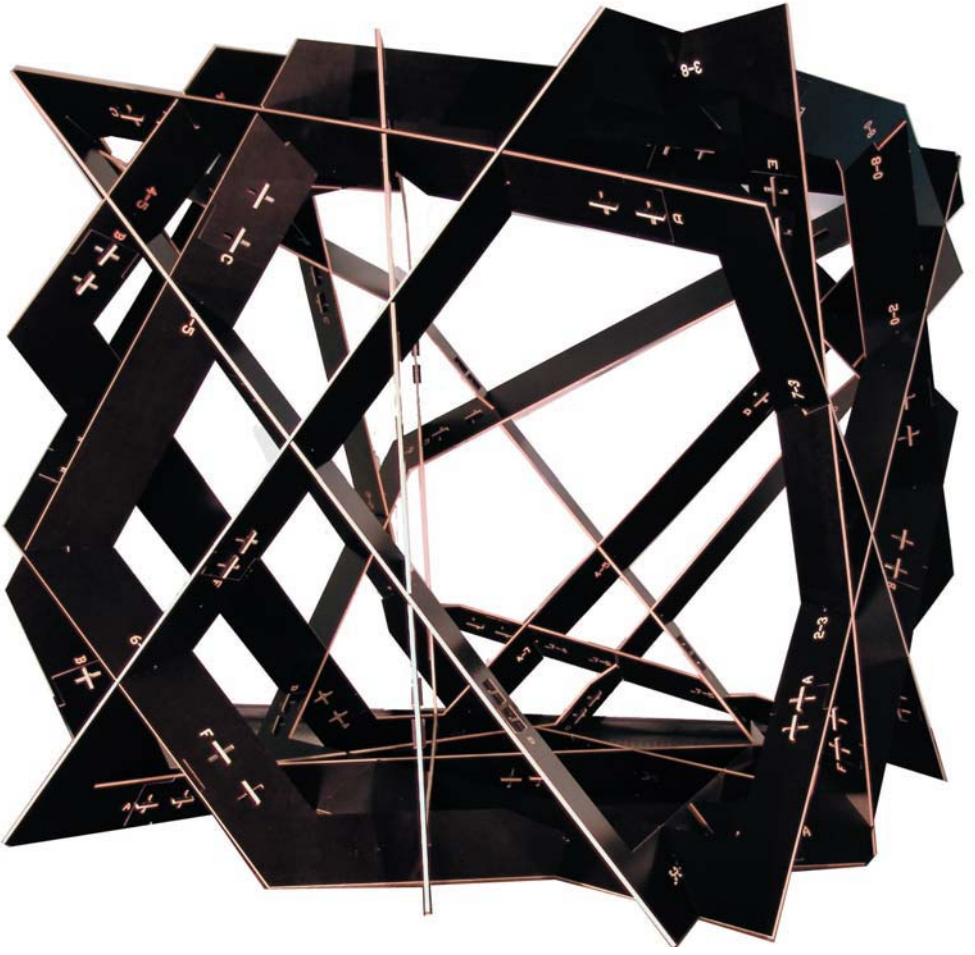
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

ETH ZÜRICH
Chair of CAAD
Prof. Ludger Hovestadt
HIL E 15.1
ETH Hönggerberg
8093 Zürich
Switzerland

www.caad.arch.ethz.ch

Layout,Graphic Design | Li-hsuen, Yeh, nds2004
Text | nds2004
Photograph | Alexandre Kapellos

Zürich, March 2005



# ACKNOWLEDGEMENTS


| | |
|---|---|
| SUPERVISORS | Prof. Dr. Ludger Hovestadt<br>Philipp Schaerer |
| From CAAD Chair | Markus Braach<br>Karsten Droste<br>Oliver Fritz<br>Russell Loveridge<br>Kai Rüdenauer<br>Kai Strehlke<br>Christoph Schindler<br>Odilo Schoch<br>Fabian Scheurer<br>Susanne Schumacher<br>Sibylla Spycher<br>Oskar Zieta |
| NDS2004 | Christian Dürr<br>If Ebnöther<br>Jörg Grabfelder<br>Anna Jach<br>Jae Hwan, Jung<br>Alexandre Kapellos<br>Irene Logara<br>Michelangelo Ribaudo<br>Hanne Sommer<br>Agnieszka Sowa<br>Detlef Wingerath<br>Thomas Wirsing<br>Li-Hsuen, Yeh |
| ETHZ Department of Architecture<br>Research / Postgraduate Studies | Bruno Dobler |
| ETHZ Betreuung Nachdiplomstudien | Anita Buchschacher |
| ETH Immobilien / Abteilung Betrieb<br>Hausmeister HI | Pascal Bisang |

# CONTENTS

# ABSTRACTS

After many communication breakdowns and linguistic misunderstandings in the group of students, a concept for a pavilion which hosts an exhibition about its genesis emerged: a computer generated and optimised structure. The outer form is a simple, clear volume. A generator defines the inner volume, subtracting it from the main structure, creating spaces, displays and tables for the exhibition. An interactive light and surface concept displays the content of the self-exhibiting prototype. All should be demountable and transportable. Due to time, program and computational restrictions which emerged during the project, the concept had to be simplified. The basic principle of the structure remained, but the integration of light, outer skin and other "extras" became non-essential decoration. The structure so became a technical, aesthetical object with no scale: complex and reproduceable in many versions. The final object is a wood-construction, whose joints are designed to be assembled without any tools and to be produced on the school's 3-axis CNC mill. Consequently, the exhibit is a prototype of the structure and not a finished pavilion. It is the result of the research of the CAAD 03/04 postgraduates students.

# 0. CONTEXTS

## 0.1 The Chair of CAAD

The Chair for CAAD at the Department of Architecture at the ETH Zurich employs current information technologies in research aimed at augmenting the contemporary concept of architecture. This extended definition includes: digital media & communications, intelligent building control & services, fabrication & building with computer controlled machines, and programmed and parametric CAAD design solutions.

## 0.2 Postgraduate Studies in CAAD

The postgraduate (NDS) studies in CAAD are open to local and international graduates with professional experience in the field of architecture and adjacent disciplines. The main focus of the course is computer based architectural design (CAAD) and its automatic production (CAM). The studies take one full year and begin in the winter term. The course is divided into a variety of modules which are taught as seminars that are concluded with an individual assignment. The conclusion to the year-long course is a group project reflecting all aspects of the research undertaken during the postgraduate course .

# 1. INTRODUCTION

An experimental structure is on exhibition. It reflects the experimentation and research undertaken by the postgraduate students in CAAD during the year 2003/2004. The design and construction drawings are not drawn by hand but are generated by the computer. The structure is programmed and parametric; many variations can be generated and produced using various CNC machines.

The manufactured result should not be regarded as an architectural statement, it does not serve a purpose, and its form does not follow a function; it is a built prototype using current information technology in design and construction. It is an attempt at using technology as a means of augmenting the capacity of designers, and not as an end in itself.

# 2. CONCEPT

The concept for the project is to use a programmed design script based on genetic algorithms to generate, optimize, and produce a complex structure. The design of the project is directly generated by a cyclical parametric generator-script. This concept then dictates that any visualized structure is but one version in a theoretically infinite series of possibilities, as defined by its given parameters and the direct intervention of the designer.

The chosen form for the structure is a cube. The inner and outer form of the structure is the result of an intersection and subtraction process. The exterior boundary of the structure is defined by its intersection with a cubic form. The (seemingly) randomly oriented crossing planes also intersect each other to create an unsystematic cellular structure. A second smaller cubic volume is then subtracted from this irregular honeycomb to create the interior space. The difference in size of the external and internal volumes defines the depth of the resulting structure.

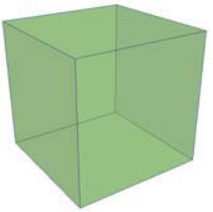
In this context the computer is not used to re-produce pre-configured forms, it is a catalyst for a new process of design. The computer becomes a tool of uncertainty in the design process, in the sense that the designer is now able to let the algorithms run and observe what kind of (unexpected) results can lead to new decisions or interventions. The resulting creative process shifts away from the traditional production of forms, to an interpretation and manipulation of the output. The resulting design is not only a play of form, but also a play of code.

This methodology is a hybrid process that capitalizes on the abilities of both the designer and the programming environment. The initial step of this computer-supported process is the creative programming of the design solution. The ensuing output, from running the design algorithm, is

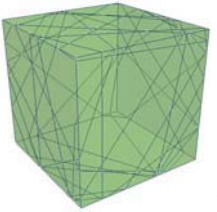
a dynamic process. Designer intervention can interrupt the process at any point to adjust parameters or finalize (or "freeze") a version of the process which best suits the given conditions. The resulting project is therefore a product of the script/program, the choice of appropriate input parameters and the direct input from the designer.

## 20 concepts for virtual to real translation

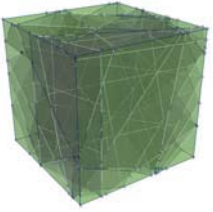
01. The structure is a concept not a fixed design
02. The structure has no scale.
03 The structure is immaterial.
04. The structure is completely defined by a logical code.
05. The code consists of parametric design algorithms.
06. The designer is able to control parameters.
07. Parameters influence the algorithms.
08. Algorithms control the process.
09. Optimization within the code guarantees the evolution of the x_cube.
10. Evolution equates to improvement.
11. Evolution is an ongoing process.
12. Design interventions can influence the process at any time.
13. A version is a frozen point in time within the process.
14. The designer can visualize a version.
15. An structure version is described through data.
16. Two-dimensional drawings are obsolete.
17. The structure can be built.
18. A physical structure version is a product of its fabrication process.
19. A physical structure version is an abstraction of the concept.
20. The structure exists.



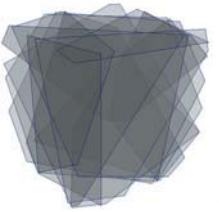
STEP 1



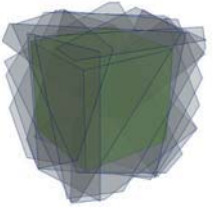
STEP 2



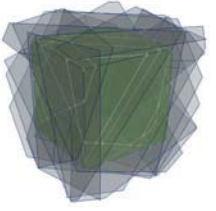
STEP 3



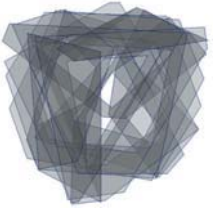
STEP 4



STEP 5



STEP 6



STEP 7

STEP 8

# 3. GENERATING

## 3.1 Overview

The goal for the structure was to develop an irregular cellular matrix where different structural densities could be generated. From this virtual matrix the exterior of the structure is defined by the intersection and boundary of a cubic volume. A second Boolean operation defines the interior space by mean of the subtraction of a second, smaller cube from the structure.

To generate this concept the script was written in Alias' Maya™ software using Maya Embedded Language (MEL). The structure, as it is generated, is not a subtractive process, but an additive one. The basic structure is formed by adding randomly rotated planar frames that are inscribed into the volume of a cube. The parameters available to the designer are the number of planes, the outer dimensions of the cube and the width of the structure (as defined by the subtraction of the inner cube). These parameters are infinitely variable.

The generator-script defines a plane using three basis points, and an orientation vector. The actual geometry of the plane is then defined by its intersection with the boundary of the cube. The resulting data is a series of Cartesian points that define the perimeter of the plane. Once the planes are defined as data, the second stage, optimization, can compare the different data values and optimize the overall structure.

## 3.2 Scripting in MEL

All scripts needed for generating, optimizing and producing the xCube were done in MEL. This scripting language allows the user to create procedures and scripts for custom modeling, animation, dynamics and rendering tasks as well as to customize the Maya interface. In this particular case it was used for calculation and visualization of a wireframe model of cube structure and than production of flat elements' drawings which were than used as a basis for milling and laser manufacturing drawings.

As scripting language for programming the xCube, scripting languages of 3 different 3D modeling or drafting applications were considered:

VectorScript (from VectorWorks),
•advantages: simple, popular in architecture offices and amongst the students of the group, possibility of customizing the user interface of the application;
•disadvantages: no NURBS, slow 3D engine;

MAXScript (from 3Dstudio MAX)
•advantages: 3D studioMAX is more popular than Maya in architecture offices; possibility of customizing the user interface of the application ;
•disadvantages: none of the group members knew MAXScript – and time was running out;

MEL (Maya Embedded Language).
•advantages: most group members had basic knowledge of MEL so programming could start at once; possibility of customizing the user interface of the application; fast 3D modeling;
•disadvantages: none known at the time;

Looking back, now that the scripting is finished, a longer and more detailed list of MEL advantages and disadvantages can be done:

Advantages of using MEL:

- no need to compile programs: scripting languages commands can be execute immediately in Maya.
- no need to exchange data: generation, optimization and production drawings can be calculated and prepared in one program.
- fast visual feedback: the results of calculation can be easily checked easily by the programmer on screen, which is important for 3D vector calculations.
- use of additional features of Maya – like springs – for next steps and experiments in optimization.
- NURBS surfaces allow fast and accurate visualization of complex forms.

Disadvantages of using MEL:

- Maya is not stable enough for executing long and complicated calculations, which were needed for optimization process.
- Some functions which could be useful don't exist in MEL, dynamic matrixes for example.

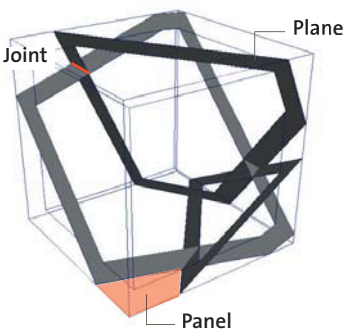
The structure of the scripts and data storage for generating and optimizing the xCube were evaluated during the whole working process. Eventually it turned out that generation and optimization could be calculated without any visualization; data output of these processes – as a text file – can then easily be loaded back into Maya. This information is particularly important for further experiments because more complex mathematical calculations are difficult to execute in Maya/MEL.

## 3.3 Generation

The American Heritage® Dictionary of the English Language, Fourth Edition
Published by Houghton Mifflin Company.

*1. the formation of a line or geometric figure by the movement of a point or line.*
*2. the act or process of generating; origination, production, or procreation.*

The outside form of the project is a cube. The structure is build by planes – flat, freely rotated construction elements of small thickness which are connected with each other that they form stable structure. Joints are located where planes intersect. The areas between planes are called panels.



## Structure of the generation script

The number of planes is given to a generating procedure as a parameter. These planes should satisfy a number of conditions: they should be generated randomly in space and be irregular – not parallel to the sides of the cube and they should be placed in space so as to intersect with the cube.

These conditions are used to set some rules for planes generating.

- Planes are defined by 3 points which are randomly positioned on the edges of the cube;
- No 3 points on the same side (pict1a).
- No 2 points on one edge (pict1b).
- The angle between a generated plane and the 3D coordinate system should not be close to 90°, in any direction (pict1c).

The information about 3 points defining a plane (as 9 values: 3 x 3 coordinates: x, y, z) is then recalculated so as to describe a plane only by 4 values: x, y, z of the perpendicular vector. Thanks to this, the data for describing a cube and redrawing it again is minimized.

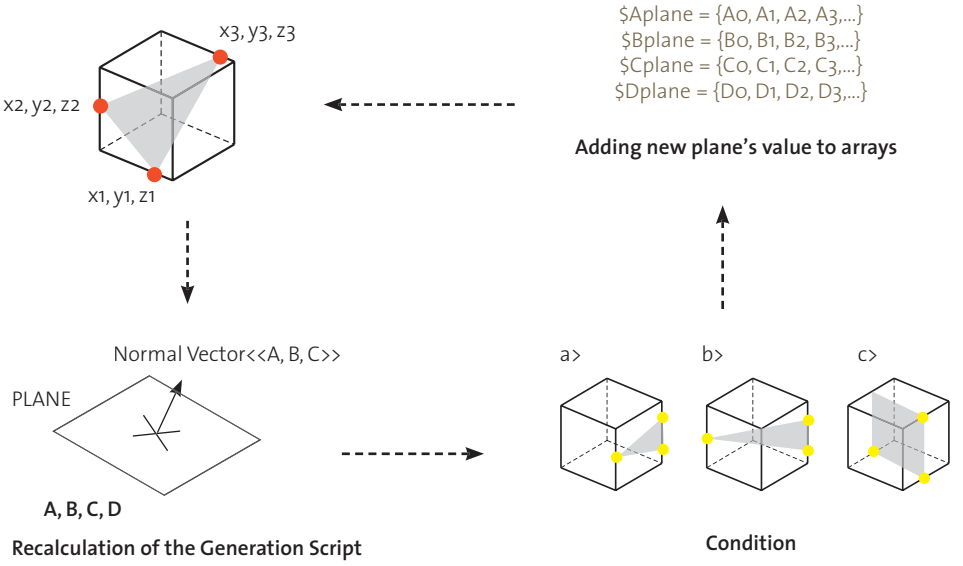
## Visualization

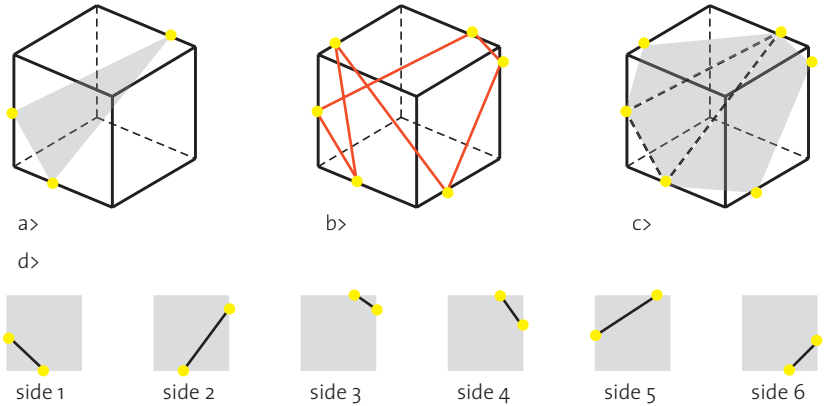
From the programming point of view there is no need to visualize data during and after the generating process. But visual analysis of the 3D model can be useful for a programmer as well as for designer. The programmer can evaluate the results of scripting and check for mistakes in the algorithm. The designer can evaluate an esthetic result and change some features by hand if needed. These are the reasons why visualization scripts were created in this phase of the project.

## Calculation and storage of data

While working with Maya we found that during file saving the variables used by scripts were not saved – so all the numeric information about the structure is lost, just a wireframe model is stored in the .mb file. From this moment on all the scripts were structured so as to be able to restore all the information (variables, coordinates, coefficients etc) only from the 4 plane arrays mentioned above. This means that it is not needed to store the Maya file with the cube model – saving the text/numeric output in plain text file is enough.

***Generation Loop - Repeat until all planes are generated***






***Method of the planes drawing.***

The 3D model of the generated structure is drawn in space with lines that connect intersection points of the planes and cube's edges. Every plane is a polygon with 3, 4, 5 or 6 edges (illustration above). The problem is how these points are connected (illustration above).

This problem is solved in the following way: intersection points between a plane and the edges of the cube are verified one after the other: every side is analyzed separately (illustration above) but drawn together in 3D space (illustration above). The idea of separate calculations for every side of the cube is used in all scripts. Due to this, most of the calculation is done on a plane – the cube's side– not in space; this makes the procedures easier and can be executed with less variables.

***Equation of a plane in space: Ax+By+Cz+D=0 (x, y, z - coordinates of a point on a plane)***

```
$Aplane = {A0, A1, A2, A3,...}
$Bplane = {B0, B1, B2, B3,...}
$Cplane = {C0, C1, C2, C3,...}
$Dplane = {D0, D1, D2, D3,...}
```

**Schema of the data storage**

```
$Aplane = {-0.8464384105, 0.595807, 0.5461631397, -0.7931392819,
-0.7075098671, 0.153747, 0.4651374461, 0.858724, 0.395076 };
$Bplane = { 0.5552307892, -0.097311, 0.4829963586, 0.4004663735,
0.8736476983, -0.950545, 0.3839430073, -1.140121, -0.981543};
 $Cplane = {-0.8580282306, -1.14713, -0.2349151238, 0.4380900804,
-0.9447640897, -1.036641, 0.391744454, -0.246476,1.199279 };
$Dplane = {0.4899769628, 0.530634, -0.5304495668, 0.0374235896,
0.6483634222, 1.036054, -0.6155449899, 0.243194, -0.22325 }
```

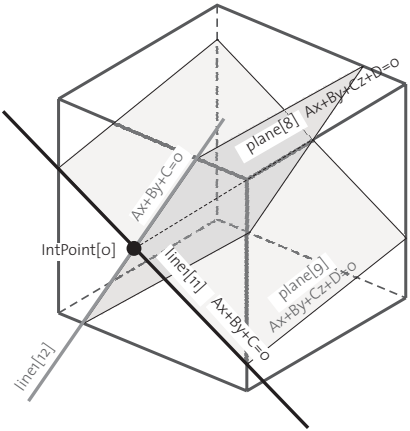
**Example - description of the study object**

The most important information for the optimization and the production of the structure is the location of the joints and how they constitute panels. Joints, from the mathematical point of view, are intersection points between lines on sides of the cube. The data about these points are stored in arrays. The data is:

- number of the cube side where the point is located.
- x ,y, z coordinates of the point.
- numbers of two intersecting lines which make this point.

Arrays are also used to store information about lines. Because the lines are always considered as being drawn on the cube's sides - they are described by an equation. Information about the lines:

- A,B,C – coefficients from the equation Ax+By+C=0 (where x, y are coordinates of points on a line).
- Number of the side where the line is located.
- Number of plane which creates the line.




| Intersection points | | |
|---|---|---|
| IntPoint[0] coordinates | | $XintPoint1[0] |
| | | $YintPoint1[0] |
| | | $ZintPoint1[0] |
| Lines | $firstLineIntPoint[0]=11 | |
| | $secLineIntPoint[0]=12 | |

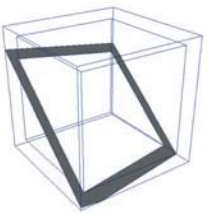
| Lines | Ax+By+C=0 | |
|---|---|---|
| line1[12] | | $Aline1[12] |
| | | $Bline1[12] |
| | | $Cline1[12] |
| | | $Pline1[12]=8 |

| Planes | Ax+By+Cz+D=0 | |
|---|---|---|
| plane[8] | | $Aplane[8] |
| | | $Bplane[8] |
| | | $Cplane[8] |
| | | $Dplane[8] |

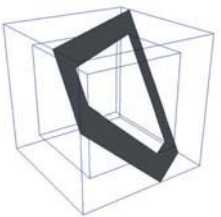
## 3.4 Variation

Each variation is defined as a 3d data model which can easily be rendered or outputed to devices such as a 3d printer or cnc machines.

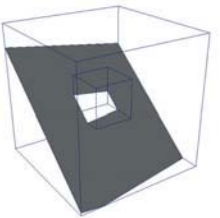
### 3.4.1 Rendering



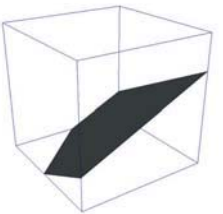
Panel: 1 / Depth: 7



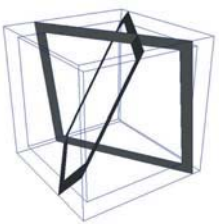
Panel: 1 / Depth: 14



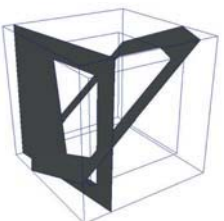
Panel: 1 / Depth: 35



Panel: 1 / Depth: 50



Panel: 2 / Depth: 7



Panel: 2 / Depth: 14

Panel: 2 / Depth: 35

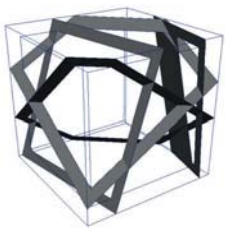
Panel: 2 / Depth: 50

Panel: 3 / Depth: 7

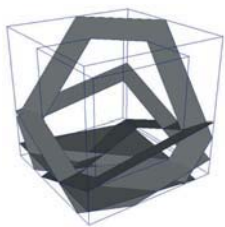
Panel: 3 / Depth: 14

Panel: 3 / Depth: 35

Panel: 3 / Depth: 50

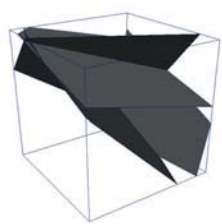


Panel:4 / Depth: 7

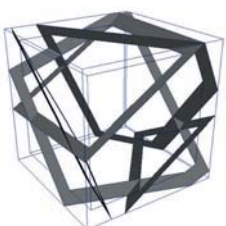


Panel: 4 / Depth: 14

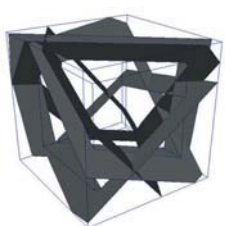
Panel: 4 / Depth: 35



Panel: 4 / Depth: 50

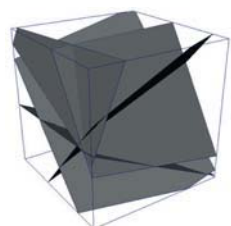


Panel: 5 / Depth: 7

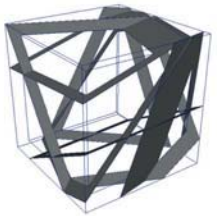


Panel: 5 / Depth: 14

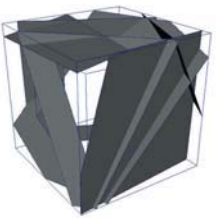
Panel: 5 / Depth: 35



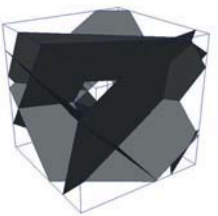
Panel: 5 / Depth: 50



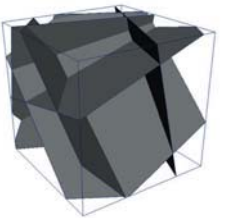
Panel: 6 / Depth: 7



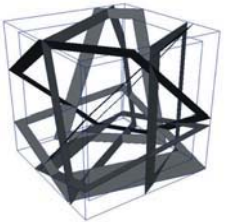
Panel: 6 / Depth: 14



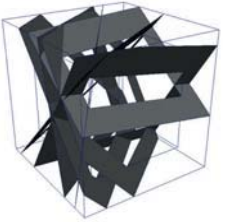
Panel: 6 / Depth:35



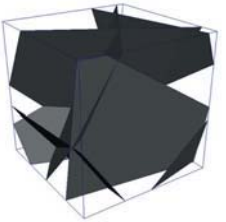
Panel: 6 / Depth: 50



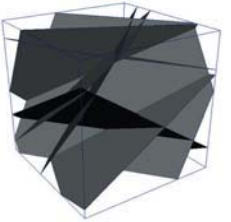
Panel: 7 / Depth: 7



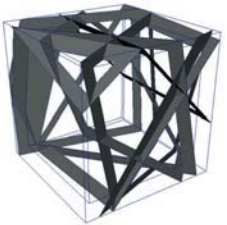
Panel: 7 / Depth: 14



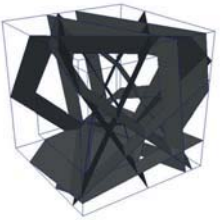
Panel: 7 / Depth: 35



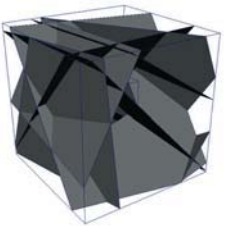
Panel: 7 / Depth: 50



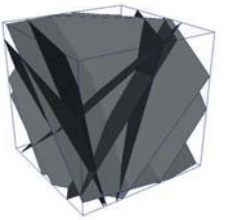
Panel: 8 / Depth: 7



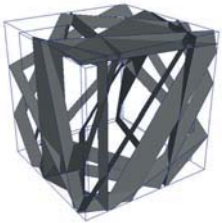
Panel: 8 / Depth: 14



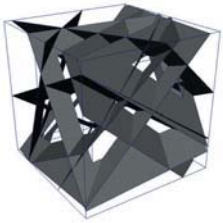
Panel: 8 / Depth: 35



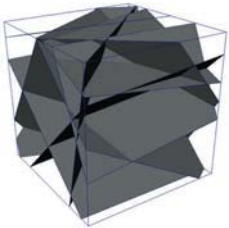
Panel: 8 / Depth: 50



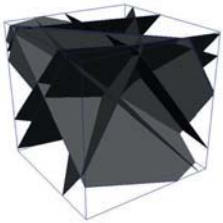
Panel: 9 / Depth: 7



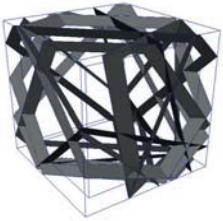
Panel: 9 / Depth: 14



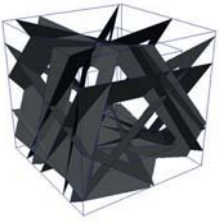
Panel: 9 / Depth: 35



Panel: 9 / Depth: 50

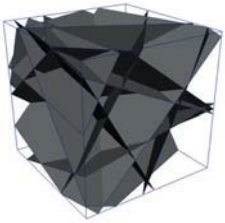


Panel: 10 / Depth: 7

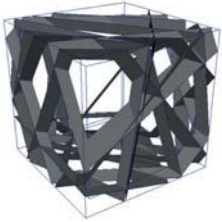


Panel: 10 / Depth: 14

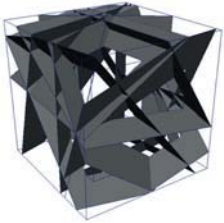
Panel: 10 / Depth: 35



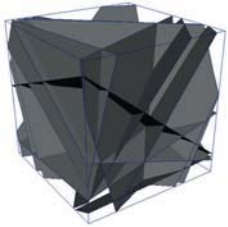
Panel: 10 / Depth: 50



Panel: 11 / Depth: 7

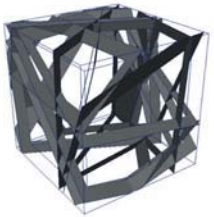


Panel: 11 / Depth: 14

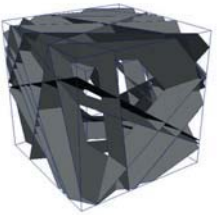


Panel: 11 / Depth: 35

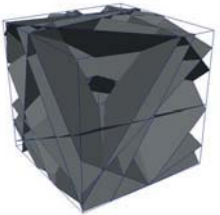
Panel: 11 / Depth: 50



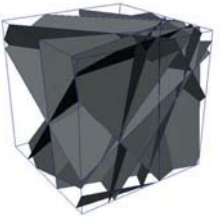
Panel: 12 / Depth: 7



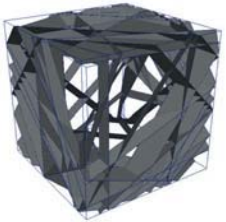
Panel: 12 / Depth: 14



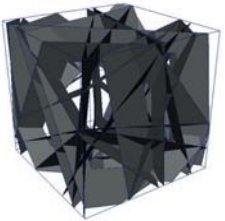
Panel: 12 / Depth: 35



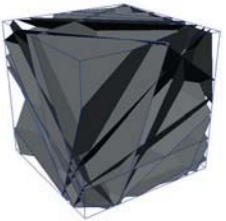
Panel: 12 / Depth: 50



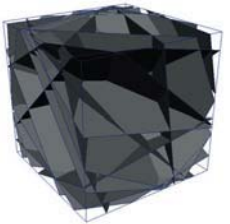
Panel: 18 / Depth: 7



Panel: 18 / Depth: 14



Panel: 18 / Depth: 35

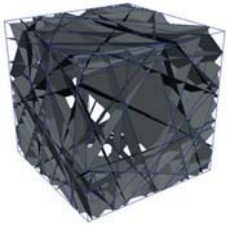


Panel: 18 / Depth: 50

Panel: 24 / Depth: 7

Panel: 24 / Depth: 35

Panel: 24 / Depth: 50



Panel: 36 / Depth: 7

Panel: 36 / Depth: 14

Panel: 36 / Depth: 35

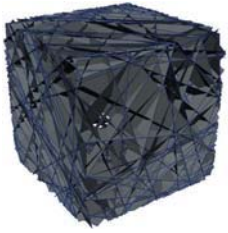
Panel: 48 / Depth: 7

Panel: 48 / Depth: 14

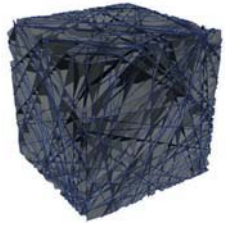
Panel: 48 / Depth: 35

Panel: 48 / Depth: 50

Panel: 60 / Depth: 7

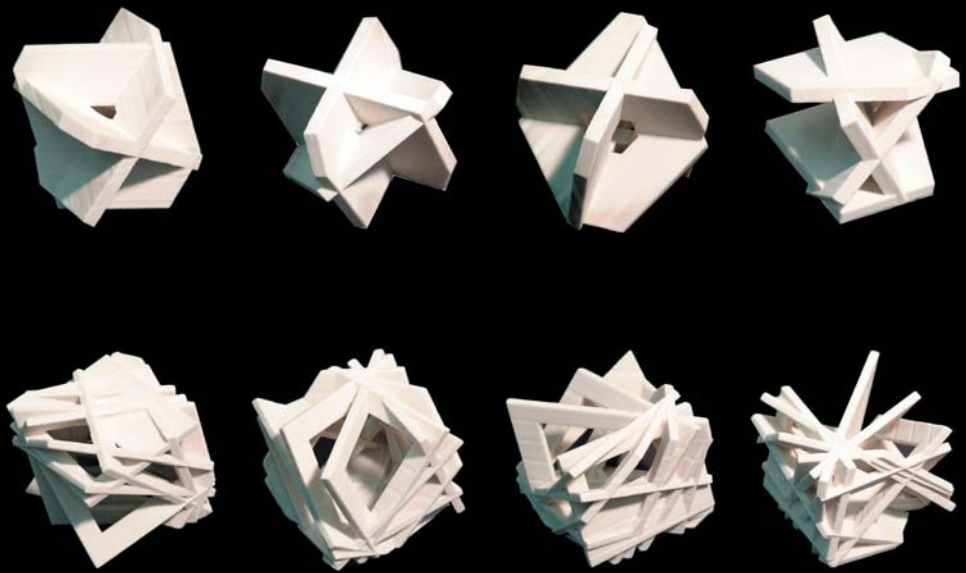


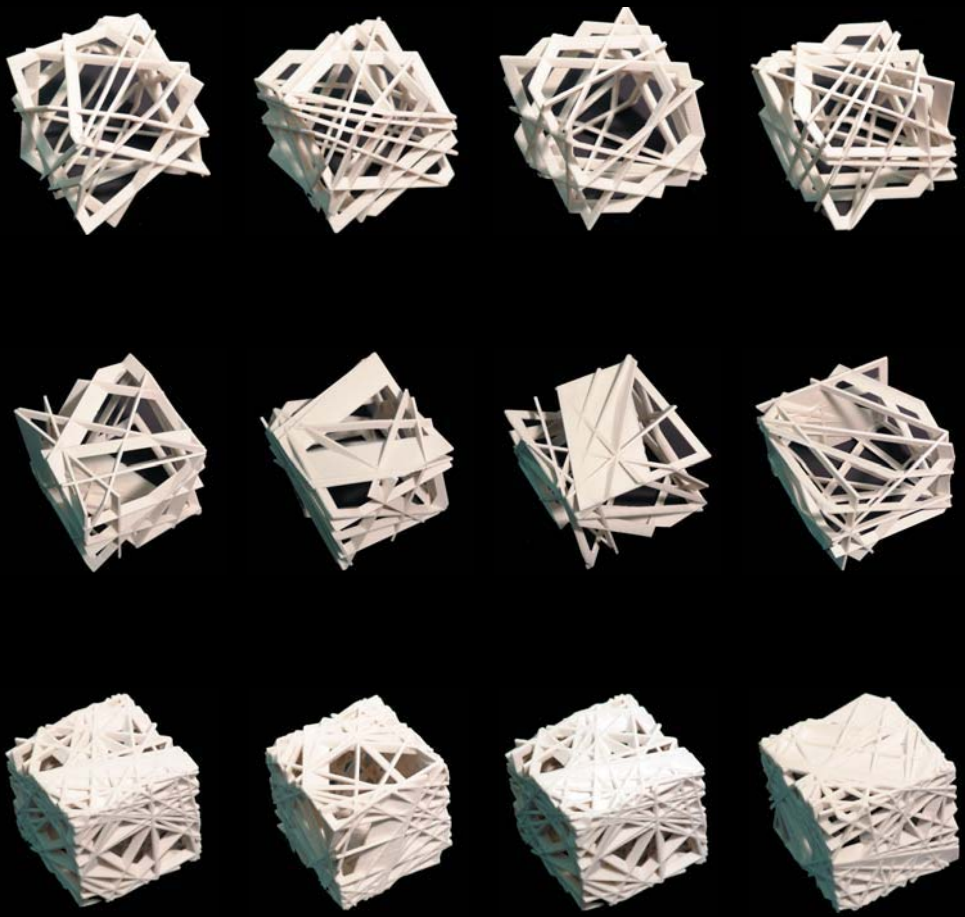
Panel: 72 / Depth: 7



Panel: 96 / Depth: 7

### 3.4.2 3D-Print





# 4. OPTIMISATION

## 4.1 Overview

Once an initial state of the structure is generated, the software then asses and optimizes the orientation and placement of the planes, according to a series of pre-defined "fitness" conditions. The optimization is a series of "test and replace" functions, where the fitness of each plane is evaluated, the worst performing plane is deleted, a new plane is introduced into the matrix, and the cycle is repeated. This form of testing is based on genetic growth principles, and is used to optimize the structure as a connected whole. The fitness tests are programmed for both aesthetic and static considerations, and finalization is only possible once the intended materiality and scale of output is known.

The conditions for fitness are:

- the planes should have a large diversity of orientation .
- the planes should not be parallel to the sides of the cube .
- there should be a minimum number of intersections between one plane and its neighbors.
- the planes should be positioned in space so that they intersect within the cubic volume.
- the length of segments between intersections should not exceed given proportions.
- there should be a minimum proportional length between intersections.

Each iteration of the structure is ranked with a overall "global fitness" factor so as to record previous permutations and compare them to the current version. The process is a computer calculated evolutionary process, which can be arbitrarily repeated, modified, and terminated manually.

## 4.2 Optimisation

*The procedure or procedures used to make a system or design as effective or functional as possible, especially the mathematical techniques involved.*
The American Heritage® Dictionary of the English Language, Fourth Edition
Published by Houghton Mifflin Company.

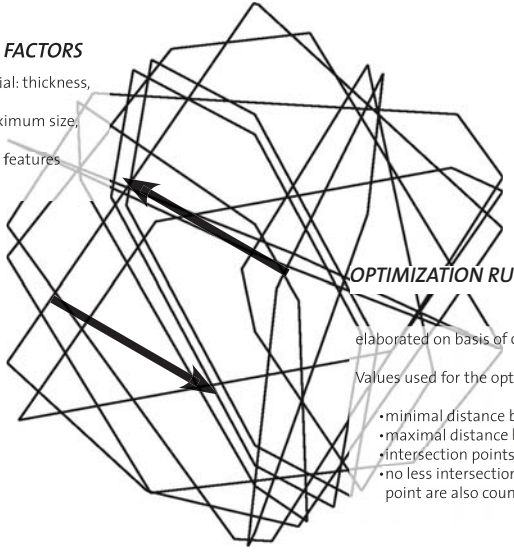
The structure generated by the MEL script cannot be used instantly as a carrying structure of the xCube because it is created randomly and it doesn't satisfy all conditions which have to be fulfilled by a structure which will be built. Generally the variety of these condition can be quite large: the type and the thickness of the material influence the structure as well as features of materials which can be used as panels. The structure is also dependent on the type of machines used to output the elements, each one having its limits and tolerances. All this information is gathered and used during the optimization of the structure.

The result of the optimization process is not a "one and only" solution, unique and always different.

The data produced by the generation/visualization script doesn't yet contain all the information needed to analyze and optimize the structure. The most important data, from the construction point of view, is how the intersection points are connected to each other – how they constitute side panels. Again all the calculations are done side by side. After this process the data is stored once again in arrays – this time every panel has its list of vertices (intersection points).

In the optimization process a simplified version of the evolutionary algorithm is used. Evolutionary algorithms are tools that mimic the natural biological evolution to produce optimised solutions to a problem. With the xCube this idea is achieved in not such a complicated way: the structure is analyzed

according to given parameters in environment of other planes. The most unfit plane is found – the one which causes the biggest number of problems in the structure. This plane is removed and a new one is generated and then the process is repeated again and again until the structure is be as close to the optimal solution as possible.



***OPTIMIZATION FACTORS***

construction material: thickness, strength, weight
panel material: maximum size, strength, weight
producing machine features
esthetic features

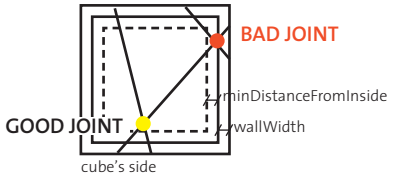
***OPTIMIZATION RULES AND PARAMETERS VALUES***

elaborated on basis of optimization features

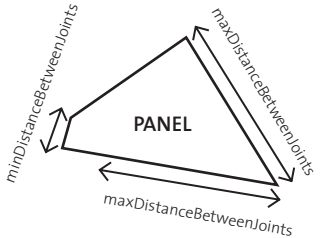
Values used for the optimization of the study object

- minimal distance between joints: 30cm;
- maximal distance between joints: 120cm;
- intersection points not closer to the edge than 20cm;
- no less intersection points than 10 (as intersection point are also counted points on edges of cube);

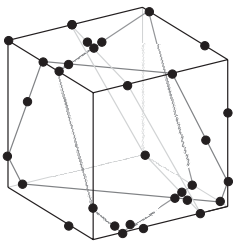
***Location of Joints***




***Distance Between Joints***




***Amount of Intersection Points***



For Every Plane

During the optimization of the instance, the following conditions are analyzed and optimized: -location of the intersection points on the side: they shouldn't be too close to the edges;

-length of panel edges: they shouldn't be longer than the maximum carrying length of the chosen material and they shouldn't be shorter than the minimum distance between joints.

-number of intersection points for every plane.

-some experiments with optimization showed that the optimization procedure has the tendency to chose as "fit" planes the ones which have no intersection points with other planes – so they don't form a stable structure.

The analysis results with a fitness value for every plane. Every time an intersection point is too close to the edge of a panel or a plane has not enough intersection points – "negative points" are given to these problematic planes. The result of the analysis phase is a list of fitness values which can be then compared:

$fitnessPlane = {-123, -256, -89, -244, -156, -289, -266, -146};

The optimization process consists of few phases:

- calculating the panel data.
- analyzing the location of the intersection points.
- analyzing the panel data (length of panels edges).
- analyzing the number of intersection points for every plane.
- calculating the fitness for every plane.
- removing the plane with the lowest fitness value.
- generating new plane.

## Designer's touch and additional tools

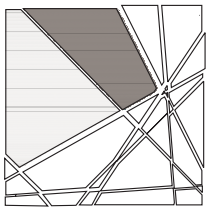
During the generation and optimization process, interaction between the script/computer and the designer is possible. Not only has the designer set the parameters for these processes, he can also make changes "by hand", tune the structure so that it satisfies an esthetic result. To make this interaction easier some tools were created. The first of them is the *printCubeArray* procedure. In Maya arrays are printed always as lists of values, without any separators or brackets. But when we want to load values from an array in Maya, a list separated by comas and in brackets is needed. *PrintCubeArray* is a small but useful tool to print arrays in an understandable format.

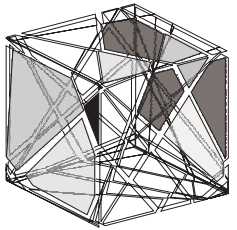
```
1
2
3
```

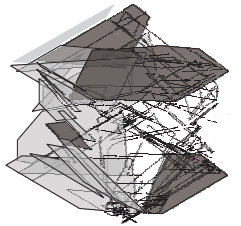
**how Maya print values**

```
$array = {1,2,3}
```

**how the values then have to be loaded.**







Another tool is the *generatePlaneAndAdd* procedure. After a cube is generated an additional plane may be needed in the structure. Adding a new plane means adding new values to the variable lists in Maya. *GeneratePlaneAndAdd* needs the coordinates of 3 points located on a plane the user wants to add to the structure. After recalculation the coefficients of the new plane are added to adequate variable lists.

Sometimes the need to analyze the fitness values of a generated structure without optimizing it (fitness calculation is included in the optimization script and is normally not printed – just analyzed and deleted from memory). The fitness procedure prints fitness values for every plane.

## Panels

The information about how intersection points (joints) are connected to each other – or how panels are formed on the cube's side is crucial for the optimization process. But it can be also useful for production: if the space between structural elements is to be filled, the shape of the cells is already calculated abd can therefore easily be extracted. The output of the *drawingPanels* procedure is a 3D model of the cube made of panels which can be deformed or used as a basis for further work.

# 4.2 Optimisation Steps

## OPTIMISATION_STEP 001

Persp

Top

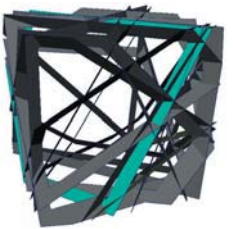
Front

Side

```
// initial cube
opt1

//arrays to draw the cube
printCubeArray $Aplane;
{-0.6852115463,-0.7426512167,0.4971806603,-0.7844700563,0.6078703267,0.303552
2522,0.5411090797,0.7513749198,0.3776949831,-0.8542221339,0.1253389532,0.89354
63422,-0.7316335487,0.5166281294,-0.1683425576}printCubeArray $Bplane;
{-0.85547919,-0.5257354989,-0.2396018465,0.3761611482,-0.6142055264,0.5274820
043,-0.6609640487,0.333143232,-0.2885789413,-0.07795086631,-0.5403049125,-0
.8925908984,0.06617009698,0.4894187047,0.17052218 79}printCubeArray $Cplane;
{0.1140953176,-1,-0.6582924576,0.56206228,-0.9704520426,0.3035417907,0.462858
2146,0.4264845831,0.2957041816,-0.9099377541,-0.5144283928,-0.5077448487,0.64
85950437,-1,-0.1753707825}printCubeArray $Dplane;
{0.590257952,1.343744045,-0.1698903816,0.3435490279,0.5576517578,-
0.6709109629,-0.2504567825,-0.8574096835,-0.2731342375,0.9169581842,0.472583
6603,0.1270492236,-0.05269316084,-0.3666677878,0.029522366 04}
source "Z:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/
fitness.mel";
fintessPlane=  -727
-801
-759
-562
-512
-681
-531
-663
-648
-740
-832        (the blue plane)
-645
-526
-712
-561      -9900 ( total fitness of the cube)
```

## OPTIMISATION_STEP 002

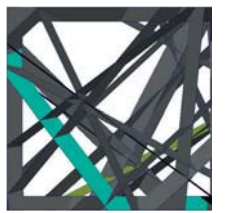


Persp

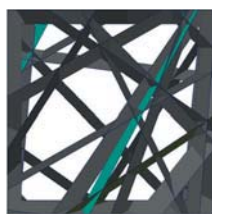
opt2 (after one step)

source "Z:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/!optimization.mel";
// Result: 1 //
printCubeArray $Aplane;
{-0.6852115463,-0.7426512167,0.4971806603,-0.7844700563,0.6078703267,0.3035522522,0.5411090797,0.7513749198,0.3776949831,-0.8542221339,-0.3514701599,0.8935463422,-0.7316335487,0.5166281294,-0.1683425576}printCubeArray $Bplane;
{-0.85547919,-0.5257354989,-0.2396018465,0.3761611482,-0.6142055264,0.5274820043,-0.6609640487,0.333143232,-0.2885789413,-0.077950866631,0.6297872424,-0.89259508984,0.06617009698,0.4894187047,0.1705221879}printCubeArray $Cplane;
{0.1140953176,-1,-0.6582924576,0.56206228,-0.9704520426,0.3035417907,0.4628582146,0.4264845831,0.2957041816,-0.9099377541,0.4192413675,-0.5077448487,0.6485950437,-1,-0.1753707825}printCubeArray $Dplane;
{0.590257952,1.343744045,-0.1698903816,0.3435490279,0.5576517578,-0.6709109629,-0.2504567825,-0.8574096835,-0.2731342375,0.9169581842,0.04683977208,0.1270492236,-0.05269316084,-0.3666677878,0.029522366 04}
source "Z:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/fitness.mel";
fintessPlane= -757
-782 (the blue plane)
-670
-572
-405
-587
-560
-655
-599
-765
-434
-535
-416
-677
-562 8976 ( total fitness of the cube)

Top

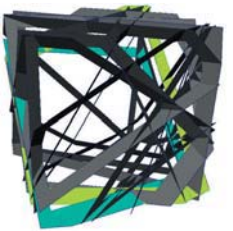


Front



Side

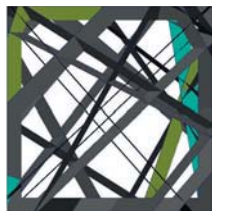
## OPTIMISATION_STEP 003



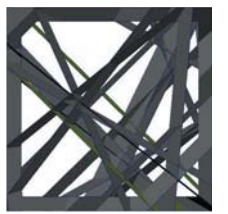
Persp

opt3 (after 2 steps)

source "Z:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/!optimization.mel";
// Result: 1 //
printCubeArray $Aplane;
{-0.6852115463,-0.830308596,0.4971806603,-0.78447005 63,0.6078703267,0.3035522522,0.5411090797,0.7513749198,0.3776949831,-0.8542221339,-0.3514701599,0.8935463422,-0.7316335487,0.5166281294,-0.1683425576}printCubeArray $Bplane;
{-0.85547919,-0.8339008479,-0.2396018465,0.3761611482,-0.6142055264,0.5274820043,-0.6609640487,0.333143232,-0.2885789413,-0.07795086631,0.6297872424,-0.8925908984,0.06617009698,0.4894187047,0.1705221879}printCubeArray $Cplane;
{0.1140953176,-0.2836257156,-0.6582924576,0.56206228,-0.9704520426,0.3035417907,0.4628582146,0.4264845831,0.2957041816,-0.9099377541,0.4192413675,-0.5077448487,0.6485950437,-1,-0.1753707825}printCubeArray $Dplane;
{0.590257952,0.8784374419,-0.1698903816,0.3435490279,0.5576517578,-0.6709109629,-0.2504567825,-0.8574096835,-0.2731342375,0.9169581842,0.04683977208,0.1270492236,-0.05269316084,-0.3666677878,0.0295223660 4}source "Z:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/!optimization.mel";
// Result: 1 //
source "Z:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/fitness.mel";
fintessPlane= -720 (the blue plane)
-707
-625
-580
-446
-594
-564
-606
-587
-712
-421
-469
-387
-683
-558 8659 ( total fitness of the cube)



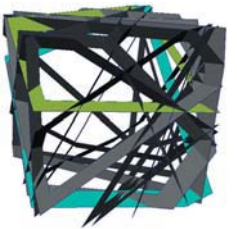
Top



Front

Side

## OPTIMISATION_STEP 004

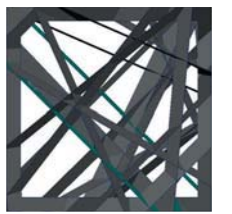


Persp

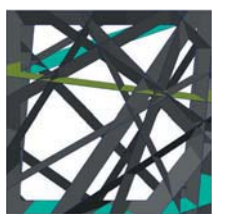
opt4 (after 3 steps)

source "Z:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/!optimization.mel";
// Result: 1 //
printCubeArray $Aplane;
{0.4127871128,-0.830308596,0.4971806603,-0.7844700563,0.6078703267,0.3035522522,0.5411090797,0.7513749198,0.3776949831,-0.8542221339,-0.3514701599,0.8935463422,-0.7316335487,0.5166281294,-0.1683425576}printCubeArray $Bplane;
{0.9370713056,-0.8339008479,-0.2396018465,0.3761611482,-0.6142055264,0.5274820043,-0.6609640487,0.333143232,-0.2885789413,-0.07795086631,0.6297872424,-0.8925908984,0.06617009698,0.4894187047,0.1705221879}printCubeArray $Cplane;
{-0.1222158881,-0.2836257156,-0.6582924576,0.56206228,-0.9704520426,0.3035417907,0.4628582146,0.4264845831,0.2957041816,-0.9099377541,0.4192413675,-0.5077448487,0.6485950437,-1,-0.1753707825}printCubeArray $Dplane;
{-0.9265281405,0.8784374419,-0.1698903816,0.3435490279,0.5576517578,-0.6709109629,-0.2504567825,-0.8574096835,-0.2731342375,0.9169581842,0.04683977208,0.1270492236,-0.05269316084,-0.3666677878,0.02952236604}
source "Z:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/fitness.mel";
fintessPlane= -595
-724 (the blue plane)
-605
-567
-494
-619
-568
-577
-599
-670
-375
-436
-369
-654
-533 8384 ( total fitness of the cube)

Top

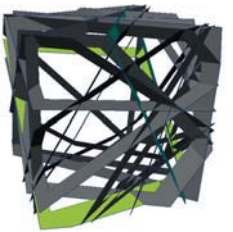


Front



Side

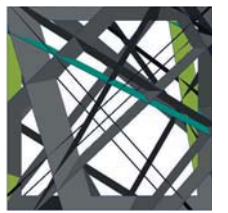
## OPTIMISATION_STEP 005



Persp

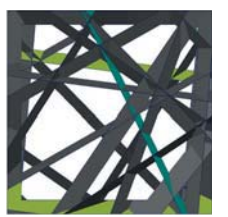
opt5 (after 4 steps) //we don t show this one on the presentation table

source "Z:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/
!optimization.mel";
// Result: 1 //
printCubeArray $Aplane;
{0.4127871128,-0.6852115463,0.4971806603,-0.7844700563,0.6078703267,0.3035522
522,0.5411090797,0.7513749198,0.3776949831,-0.8542221339,-0.3514701599,0.89354
63422,-0.7316335487,0.5166281294,-0.1683425576}printCubeArray $Bplane;
{0.9370713056,-0.85547919,-0.2396018465,0.3761611482,-0.6142055264,0.5274820
043,-0.6609640487,0.333143232,-0.2885789413,-0.07795086631,0.6297872424,-0
.8925908984,0.06617009698,0.4894187047,0.1705221879}printCubeArray $Cplane;
{-0.1222158881,0.1140953176,-0.6582924576,0.56206228,-0.9704520426,0.30354179
07,0.4628582146,0.4264845831,0.2957041816,-0.9099377541,0.4192413675,-0.50774
48487,0.6485950437,-1,-0.1753707825}printCubeArray $Dplane;
{-0.9265281405,0.590257952,-0.1698903816,0.3435490279,0.5576517578,-
0.6709109629,-0.2504567825,-0.8574096835,-0.2731342375,0.9169581842,0.04683
977208,0.1270492236,-0.05269316084,-0.3666677878,0.02952236604}
source "A:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/
fitness.mel";
fintessPlane= -535
-655
-654
-558
-447
-568
-589
-605
-618
-665
-373
-476
-389
-679 (the blue plane)
-551 8362 ( total fitness of the cube)
.



Top

Front



Side

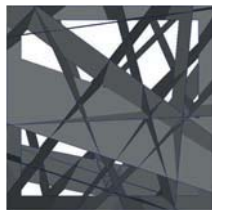
## OPTIMISATION_STEP 300

Persp

opt300 (after 300 steps)

printCubeArray $Aplane;
{-0.6861576864,-0.8967815676,0.1024927641,0.90153727 46,-0.06723860033,0.3505639473,0.2962542798,0.7513749198,0.3776949831,0.0374960806,0.06417568244,0.8935463422,-0.7316335487,-0.1120902461,-0.1683425576}printCubeArray $Bplane;{-0.2348370697,-0.2310051497,-0.07631474529,-0.6997724918,-0.3337485 83,0.4734546185,0.9238132462,0.333143232,-0.2885789413,-0.8458310673,0.1887016771,-0.8925908984,0.06617009698,0.8091956854,0.1705221879}printCubeArray $Cplane;
{-0.5787754168,0.5612787313,0.1664231234,-0.3754748303,0.8293387158,-1,0.07133735348,0.4264845831,0.2957041816,-0.5386698832,0.9153831063,-0.5077448487,0.6485950437,0.04881745535,-0.1753707825}printCubeArray $Dplane;
{0.397131201,0.1730708389,-0.05191477092,0.06890150668,0.05356957612,0.4764696504,-0.3031986495,-0.8574096835,-0.2731342375,0.8913811477,-0.9955261385,0.1270492236,-0.05269316084,-0.7323383303,0.029522366 04}
source "A:/___CONTENT___/GEOMETRY/090418GEOMETRY_COMPLETE/opt/fitness.mel";
fintessPlane= - 496
-429
-431
-632
-508
-415
-515
-431
-391
-578
-319
-618
-528 -6291 ( total fitness of the cube)

Top



Front

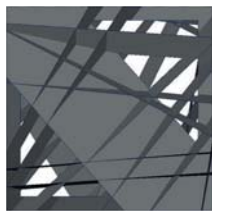
Side

## OPTIMISATION_STEP 3000

Persp

opt3000 (after 3000 steps)

printCubeArray $Aplane;
{-0.6840459908,-0.9169479293,-0.09319688926,-0.1826128985,-0.3446565603,-0.08220094687,0.5166281294,0.0374960806,0.37769949831,0.06417568244,-0.03658594423,-0.1683425576,0.3165069806,-0.9538141297,0.615628249}printCubeArray $Bplane;
{0.5337042875,-0.8163656739,-0.8516940395,-1,-0.2873557116,-0.1073388793,0.4894187047,-0.8458310673,-0.2885789413,0.1887016771,-0.1463959257,0.1705221879,-1,0.143320981,-0.67433088}printCubeArray $Cplane;
{-0.5378307192,-0.5864130723,0.08146179449,-0.4671026336,-0.1940068182,-1,-1,-0.5386698832,0.2957041816,0.9153831063,0.3599932206,-0.1753707825,-0.1950670765,-0.1677668147,0.3012193158}printCubeArray $Dplane;
{0.3189677126,1.334977887,0.8234974581,0.9392271677,0.4256135625,0.9007125179,-0.3666677878,0.8913811477,-0.2731342375,-0.9955261385,0.01122602639,0.029522366604,0.08326195688,0.9305417502,-0.5017104259}
fintessPlane= -296
-383
-435
-413
-392
-279
-533
-641
-488
-408
-440
-402
-359
-482
-425 -6080 ( total fitness of the cube)



Top

Front

Side

# 5. MATERIAL and ASSEMBLY STUDIES

## 5.1 Overview

Every prototype was outputted to different CNC machines available at the Chair for CAAD at the ETH in Hönggerberg. Small prototypes were plotted on a 3D printer (additive process), as well as on a lasercutter. For the manufacturing of the structure, a 3-axis mill (subtractive process) was used.

## 5.2 Approach

For the development of the building system the following aspects were relevant:

- Ease of use: easy assembly without tools, transportability and storage (light, small elements).
- Manufacturability: rapid manufacturing on the chair's mill, no postprocessing (sanding), reasonable material costs.
- Aesthetic aspects.

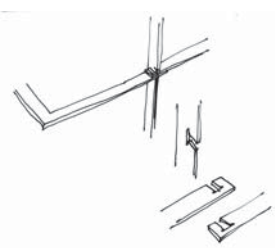
These outlines subsequently influenced all other decisions in the process of finding appropriate building solutions.

## 5.3 Structure

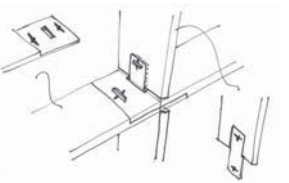
Evaluation of different principles for building the structure.

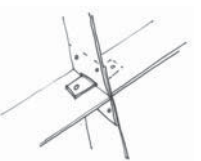
The intial solution for building the structure was found intuitively and rapidly: the elements have slots which allow them to be slotted one into the other. The only problem with this approach was that some boards had several slots which are not parallel to one another, this led to the development of the speical detail (see slot joint). At a much later stage in the process the building system was questioned again, because we needed to subdivide the elements fo be able to manufacture them so it might have made sense to subdivide all the elements. The final solution is very close to the intital idea.

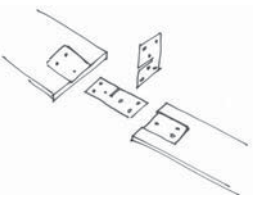
*system with only one joint - every intersection is a connection*



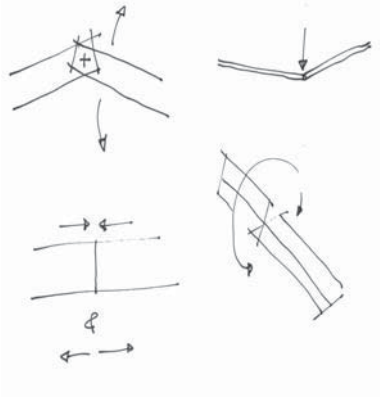
*system similar to the one used in the NDS 2002 Pavilion using sheet metal plates*







*Various stresses on the elements*

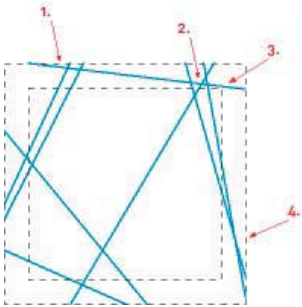


*Material evaluation*

| MANUFACTURER | MATERIAL | FINISH | THICKNESS (mm) | SHEET SIZE(m) | Fr./m2 | kg/m2 | VERDICT |
|---|---|---|---|---|---|---|---|
| Argolite | solid core | white | 8 | 2.6*1.3 | 80 | 11.2 | sheet too small |
| Küchler | solid core | white | 8 | 2.75*2.04 | 56 | 11 | sheet too small |
| Badertscher | Trespa | - | 10 | 3.65*1.86 | 101_- | 14 | too expensive |
| Maagtechnik | Polypropylene | white | 8 | 2*1 | - | light | too bendy |
| Küchler | Multiplex Okumé | natural | 10 | 3.1*1.83 | 24.1 | 5 | don't like wood finish |
| Sax-Zim | Multiplex Phenol | brown | 12 | 3.05*1.52 | 40.9 | 8.5 | sheet too small |
| Sax-Zim | 3-ply | white | 16 | 5.2*2.1 | 36.5 | 7.3 | mot ideal for milling!! |
| Küchler | Osb-Agepan | chips | 10 | 2.5*1.25 | 9 | 6.5 | mot ideal for milling!! |
| Küchler | MDF | white cover | 8 | 2.8*2.07 | 11.9 | 6.8 | sheet too small |
| Küchler | MDF | white cover | 16 | 4.2*2.07 | 17.4 | 13 | strength |

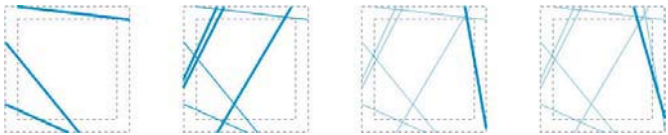
## 5.4 Ideas on Programming

The development of the generating/optimising code and the way the structure is actually built went hand-in-hand. Here are a few examples:




***Potentially problematic situations:***

*1. very flat angles close to the edge*
*2. small cell size*
*3. elements wholly in one wall*
*4. flat intersections*



***Idea on how the assmebly order could be determined:***

*1. choose the maximum amount of elements which do not cross each other on any side, these are the first ones assembled.*
*2. repeat the procedure for remaining elements.*

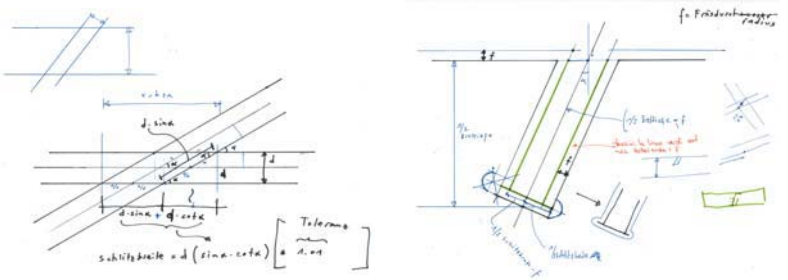
***3D print with difficult situations marked***

## 5.5.1 Slot Joints

The slot joint occurs whenever two boards/elements intersect. Because all the intersection angles are different, the actual width of the slot had to be generated, each being diiferent. The sketches below show the approach, the parameters for the calculation of the slot width are:

- thickness of material
- angle of intersection
- diameter of the milling bit

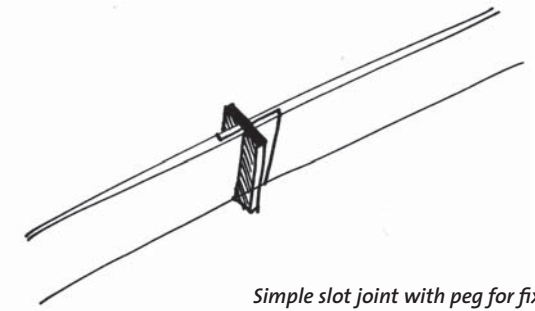
In this way it was possible to generate parts of the milling path.



*Sketches describing the geometry of the intersection slot*

*For the assembly, both parts numbering and special details had to be introduced*

## 5.5.2 Pocket Joints

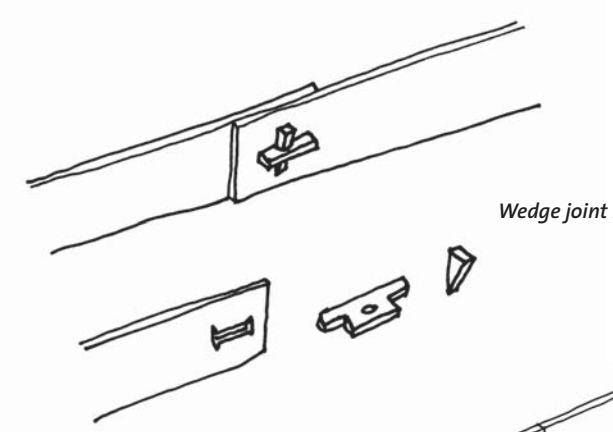


*Simple slot joint with peg for fixing*

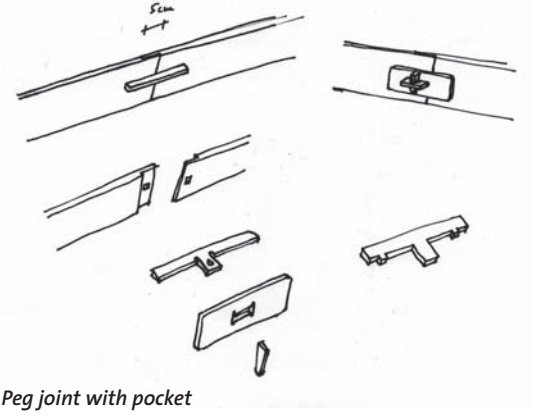
The pocket joint exists for two reasons:

1. In order to be able to assemble the structure, the frames had to be divided.

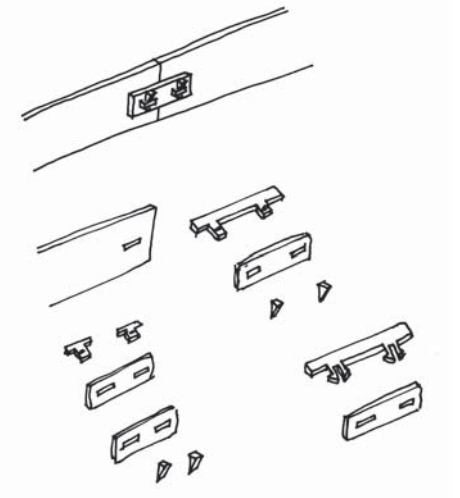
2.The size of the milling table and the size of the chosen material made it impossible to manufacture the elements in one piece.



*Wedge joint inspired by rural furniture*

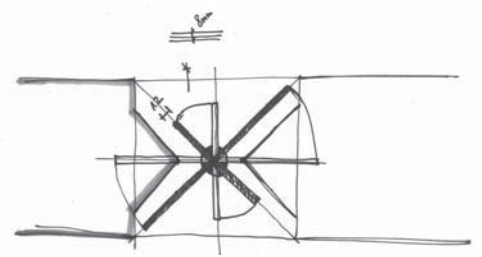



*Peg joint with pocket*

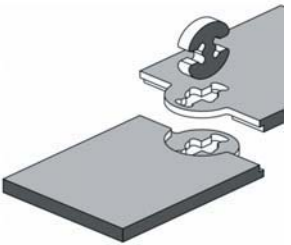


*Double peg joing, alternatively with clicking connector*

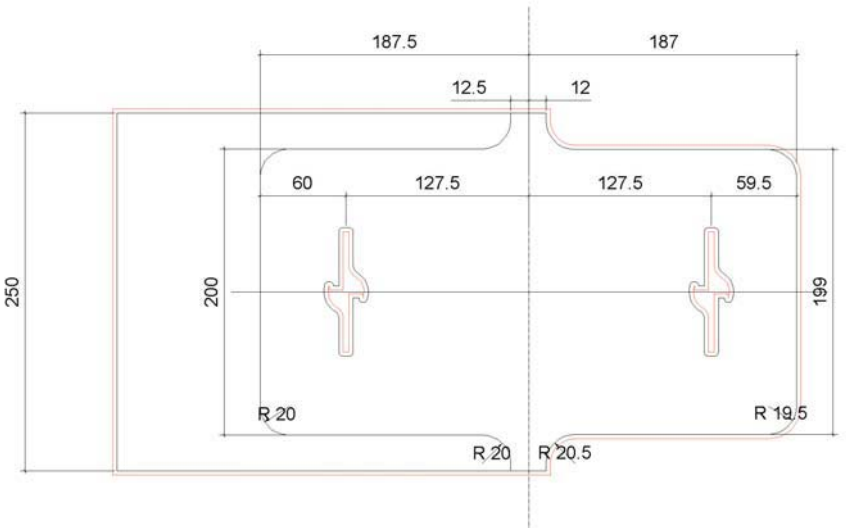
*Overview over different types of connections*



*Drehkreuz*



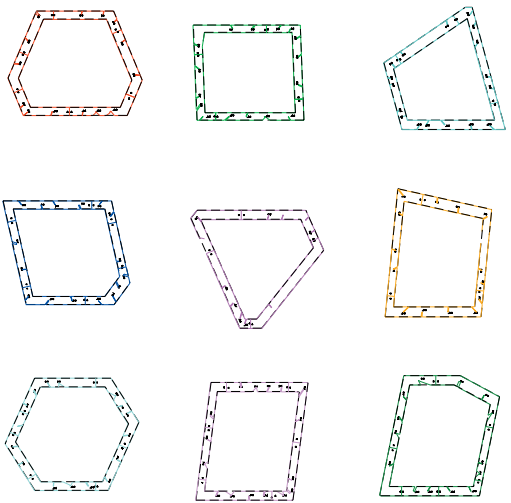
*Self locking joint from c_labor offenbach*



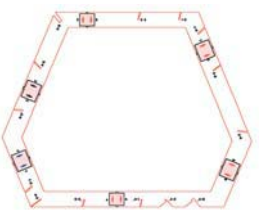

*Measured drawing of the pocket joint showing the tolerances*

## 5.6 From Code to Mill

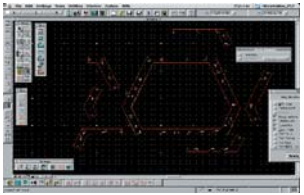
Although theoretically possible, the programming did not deliver finished data ready for production due to time limitations. A number of steps had to be done manually or semi-automated using various software packages to finally generate the code for the mill.



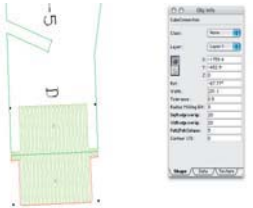
*Output from script*



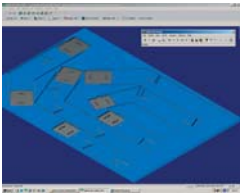
*Finished frame with all details ready for milling*



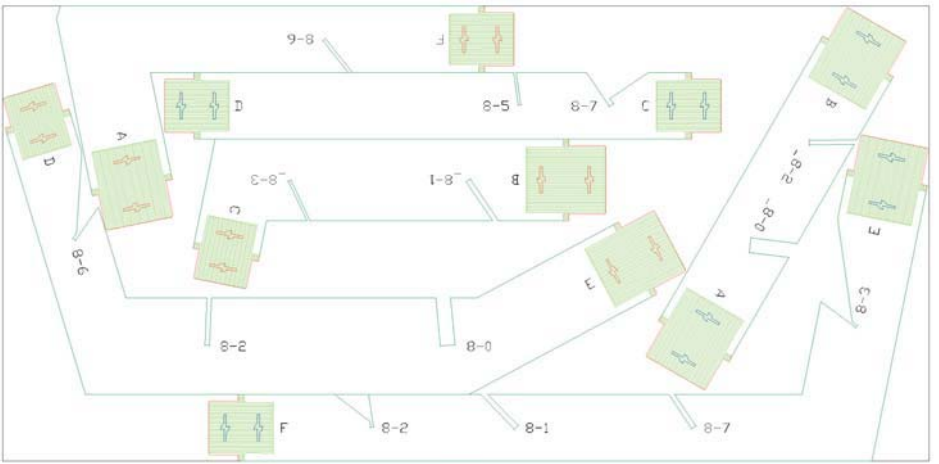
*Import into Microstation to separatte the elements*



*Parametric connection detail in VectorWorks (plugin with tolerance factor for the mill)*



*Milling parameters are set in SurfCam*




*Final detailled plate with pieces for one frame*

# 6. Automated Construction Drawings

## 6.1 Overview

All frames and CNC joints should be automatically milled out of planes. This workflow demanded the preparation of a layout for milling. A third part of the script finalizes this task. The needed slot joint is projected onto each frame, each of these is extracted from the optimised 3D model obtained in the preceding stage. This draft is then used as a plan for the manufacturing of the structure. The size- and transport-conditional cnc-joints will additional retrofit manually.
The draft will subsequently be translated into an engine-understandable code with SurfCam and transfered to the 3-axis mill as milling paths.

## 6 .2 Programming Approach

Because frames are flat (the material thickness is 1/200 of the height of the cube) and are to be manufactured by a 3d-axis milling machine or laser cutter: the frames are always considered and calculated as 3D wireframe drawings without thickness.

A method of calculation and basic rules for assembly were prepared: the task was to write the procedure which draws the joint and then connect this procedure to the general script structure so that the joints could be drawn automatically for wach intersection point. Moreover, the assembly order should be taken into consideration. During the preparation of construction drawings the whole structure becomes less and less abstract, more information about design, materials and the manufacturing process is needed: the width of the wall, the thickness of the material, the milling bit size (if the mill is going to be used).

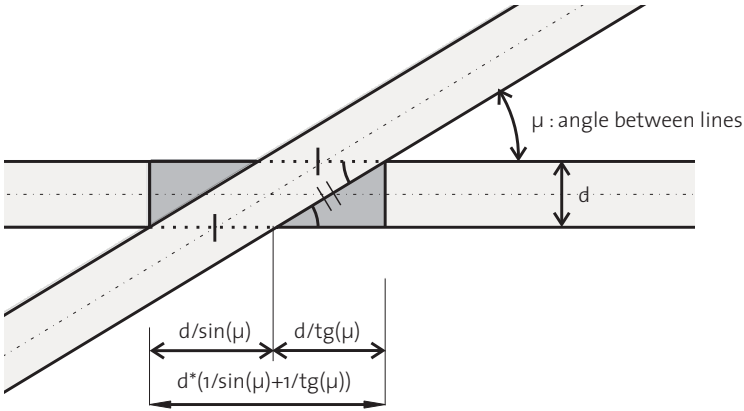
Inputs for automated construction drawings are:

- the wall width; the wall width is virtual since no planes really have this width, because every one of them is rotated in space.

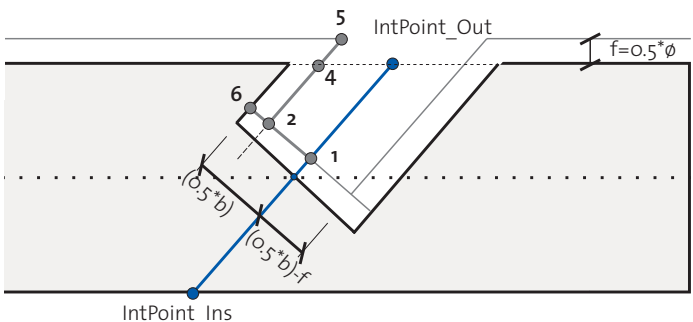
- the thickness of material used – the slot width depends on this value.

Joint width calculation is described in the illustration on the right page. The width depends on the thickness of the material used and the angle between intersecting planes.

*The calculation of slot width*




*How to describe the slots by script*

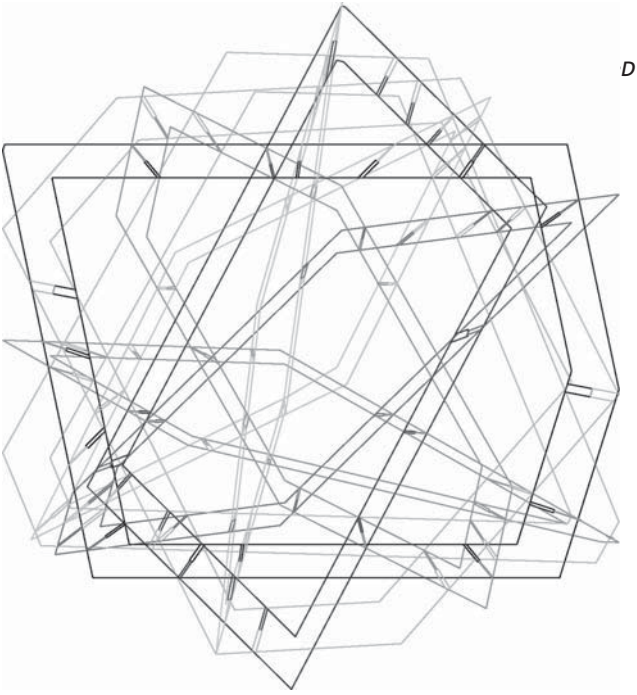


## Method of assembly:

The information about planes is stored in an array, so the planes have their own ID numbers (from 0 to 8). Every operation and calculation in the scripts uses these numbers to identify planes. It was decided that it could also useful to set the assembly order to be the same as the order of the planes in our project.

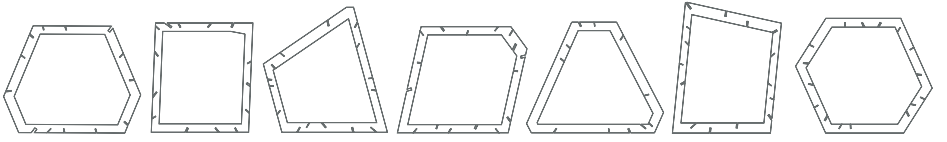
The idea about the assembly is that the first element on the cube's side has only outside slots – also all other elements which are connected with this particular element only have inside slots. By outside slots we mean slots that face the outside of the cube. Inside slots are slots which face inside of the cube.

As it was mentioned above, the assembly order is the same as order of the planes in the xCube. The slots are automatically drawn on every plane in. The frames and slots are put on separate layers – one for every plane and are rotated in space. A flat drawing is then created. These drawings are then used as a basis for production.



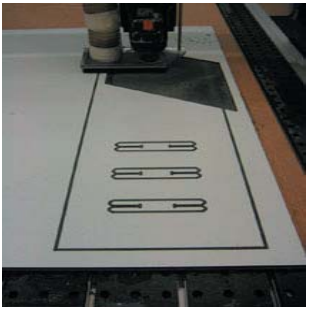
*D model with slots*

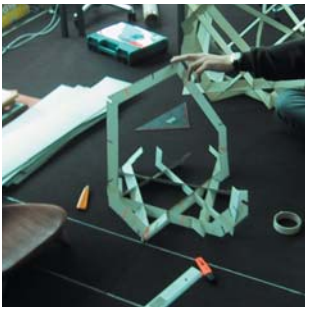
*Flat drawing for every frame*

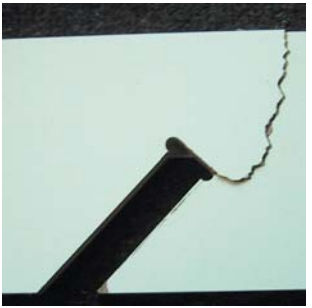


# 7. FIRST PROTOTYPES

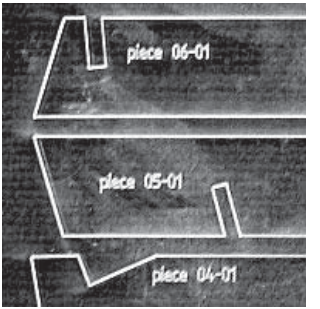

(001)Joerch.mcd
PWR 100.0 0:01:27
SPD 5.0 COPIES 0
PPI 500 IMG DEN 5

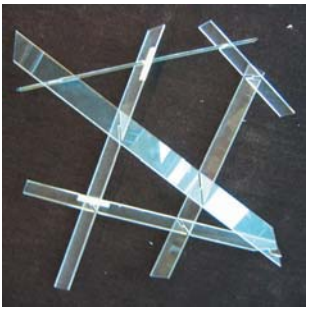




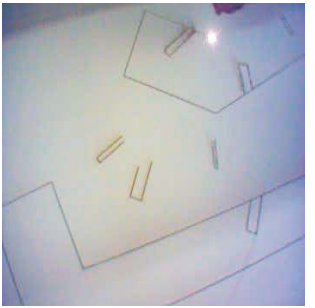


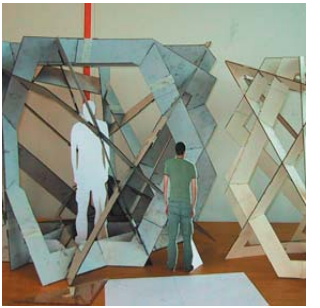

ESIGNER


piece 06-01
piece 05-01
piece 04-01

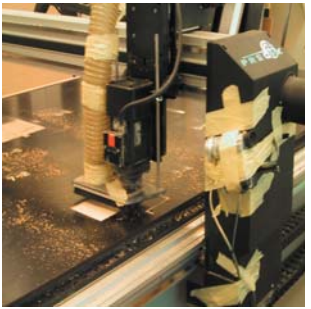
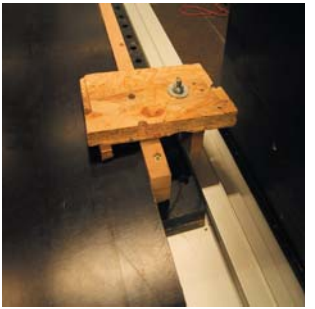
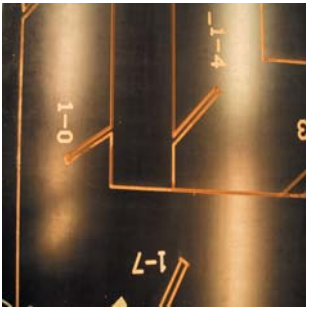
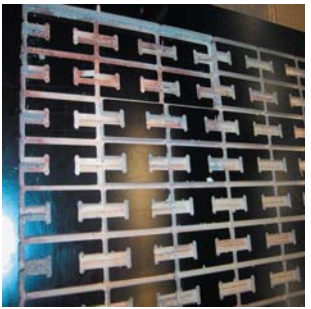
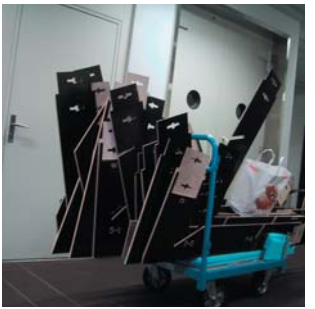



toe tight
toe tight

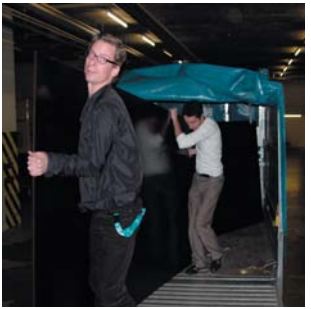


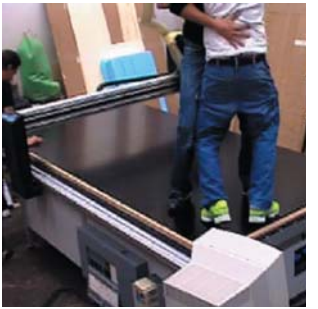
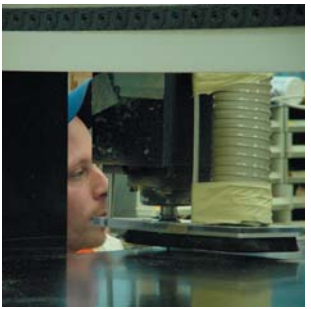
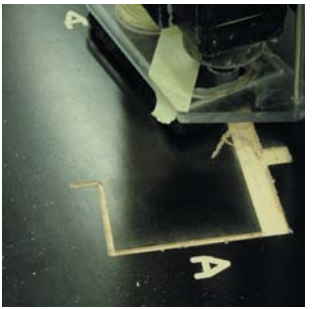


# 8. PRODUCTION and ASSEMBLY




1-4
1-0
1-7










A
A


3-8

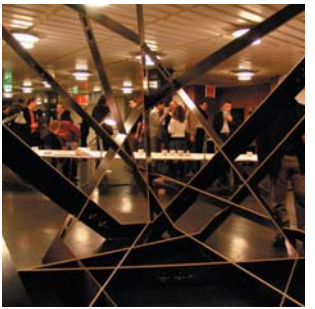
# 9. INAUGURATION

*INAUGURATION*

*The xCube was presented to the public on October 14th 2004 at the ETH in Hönggerberg.*



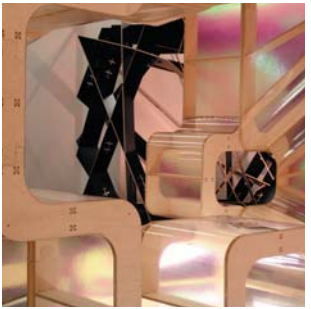






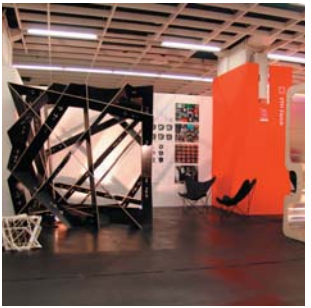
*EXHIBITION*

*The xCube and the ESG_Pavilion (nds 2003) were exhibited at the Köln imm Furniture Fair in January 2005. Both pavilions were awarded the prize for innovative architecture and design by the FuturePoint jury.*









# 10. CONCLUSION

The challenges of this project were many and not restricted to technical or design issues alone. The differing opinions amongst the group members about design and conceptual ideas, the varied levels of technical expertise and the diverse fields of interest all influenced the final work. These differences provoked critical discussions about programmatic issues that arose during the project, but more importantly, about the greater scope of Computer Aided Architectural Design.

The methodology used shifted the group's pre-occupations away from the design aspect, away from a specific architectural solution to a more generic and parametric oriented one. The result is therefore a procedure yielding an infinite series of designs defined by parameters and designer input.

The effort to create the generating script was considerable. The suitability of the output was only as good as the programming solution and the skill of the programmer. The software could not be expected to deal with "real world" architectural design in all of its complexity, and should only be seen as a tool to augment the architectural process. Due to these limitations, the current concept for the xCube is clearly conceptual or theoretical architecture.

The final prototype of the xCube was fabricated and assembled in a 48 hour period. The complexity of the irregular structure, the manufacturing of the parts, the incorporation of the connecting joints and the logic of the assembly were only possible through the use of computer assisted technology. The limitations of the script could be minimised with additional expertise in programming and a clearly defined architectural brief. The ability to generate and fabricate additional versions of this design is clear, and each subsequent generation, fabrication and assembly cycle would only continue to be more and more efficient with each new version.

# APPENDIX:

## ADDITIONAL STUDIES

### A.1 Interactive Light Study

### A.2 Surface Lasering and Milling Studies

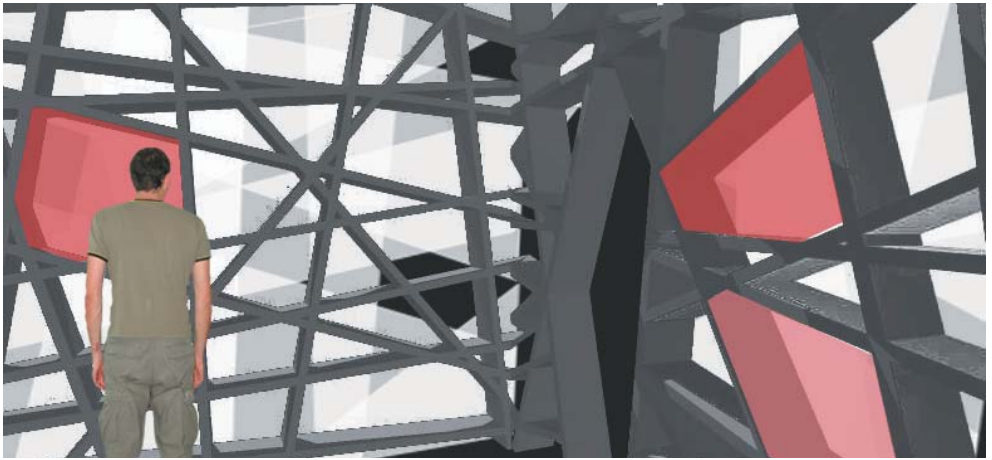
# A.1 Interactive Light Study

## Concept

The initial idea was to build a pavilion that exhibits the diverse processes viewed during the year. The structure was to be a container for different *sub-processes*: rapid prototyping, milling, laser-cutting, interactivity, building IP, interface, configurator.

The idea to focus on light is nothing new. Light has always been part of the architectural process. So why interactivity? The final project was seen by the group as an ongoing process, constantly evolving. The final built instance would have to show this process, showing the result as being a keyframe in an endless sequence.
But how does one translate this unfixed state? How can it be represented? Interactivity was the groups answer.


Interactivity was a theme that was present from the beginning in the group discussions: «pro-active» and «reactive» as opposed to static. It was also the opportunity to use technology that otherwise would not be accessible to us: network controlled devices, sensors, etc.



## Proposals

Various ideas and concepts where discussed and tried-out before reaching a final proposal. It is important to mention that as the structure of the pavilion (and our perception of it) evolved, as did all of the "sub-processes": texture study and light concept.

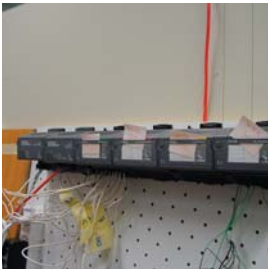
First approach: the pavilion as a unified object. A single light source inside the construction projects the structure outwards, revealing it through an outer skin. A variation in intensity and colour gives the pavilion different moods and atmospheres.

Second approach: the honeycomb cells of the structure are used as light boxes. Lasered coloured plexiglass panels containing information are positioned at eye-level. Distance sensors detect the presence of visitors or movement and turn on and off the light inside the light box, following the person as he/she moves. At this point, the texture study and the light concept evolve together, each effecting

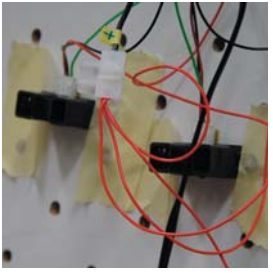
on the other. The exhibition content is highly visual and tactile, a combination of light and sculpted surfaces, becoming an intensive experimentation in textures and its interaction. Different types of lighting and bulbs are tested in co-ordination with a variety of Plexiglas colours and types: transparent, translucent, opaque.

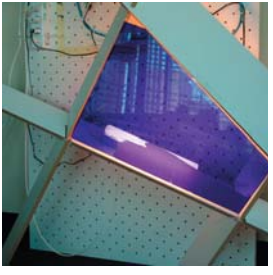
Final approach: the light & surface concept evolves from being a *skin* project to a *screen* project. The group decided to leave the outer aspect of the structure "as is", without add-ons of any type. Clean and simple, the structure speaking for itself.

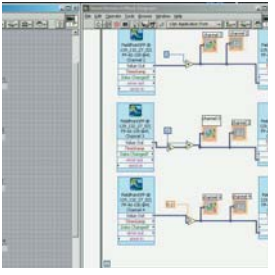
A Plexiglas panel is inserted inside a frame, giving it back its planar aspect. This panel is broken up into smaller pieces, following the projection lines of the surrounding frames, creating fractures. Onto the surface, a mixture of keywords, mel-scripts, sketches are lasered at different depths and intensities. One option is to have lights placed behind the screen revealing the texture, giving the whole surface a three dimensional effect, while another idea is to have fluorescent lights placed directly behind the fractures. The lights are controlled by distance sensors appropriately placed and interact with the visitor. In a way the project becomes a low-tech lcd screen




OSRA
SRAM







## Technology

Various tools and instruments were available to us:
The Barix Barionet network enabled automation interface is a small unit that allows the user to control a number of inputs and outputs (analog and digital) through a web interface. In a simple set-up, it allows for the control of electrical devices over a network (local or internet). A basic mock up was set up for experimentation, and a simple flash interface was programmed that controls the on/off switch of a bulb. Another button put the light into blicking mode.

The set up for this unit is basic but not very versatile and provided little interactivity and limited connections/lights.

As the concept for the light and surface evolved so did our needs to control the instalation and to improve its interactivity. The hardware provided by National Instruments (Field Point) allows for more interaction, but through a much more complex procedure, thus providing more flexibility. A programmable interface (through the software LabView) allows the control of input and output modules. In our experiment, distance sensors (input) modify the voltage fluorescent lights (output) receive, dimming them or increasing their brightness.
The clear advantage of this solution is that the software allows for loops, conditions, booleans, time-outs and that scenarios can be thought up.

# A.2 Surface Lasering and Milling Studies

## Goal

The development of NDS Pavilion was based on four guidelines:

Geometry and construction optimization studies (both using computer software like Maya and Vectorscript for Vectorworks), construction material research (traditional testing of materials under stress) and finally, surface lasering and milling studies (using table laser and CNC-milling machine).

The aim of the exploration was to find a material that acts as an information carrier and shows the development process of the pavilion.

Premises were:

1. surfaces should be opaque enough to veil the construction behind but reveal the lines of construction as shadows when illuminated from inside.

2. materials should resist mechanical constraints.

3. materials containing PVC(Polyvinylchloride), which cannot be processed on the laser (dangerous vapors) and other poisonous ingredients should be avoided, detailed information about material content was taken under consideration in any case.

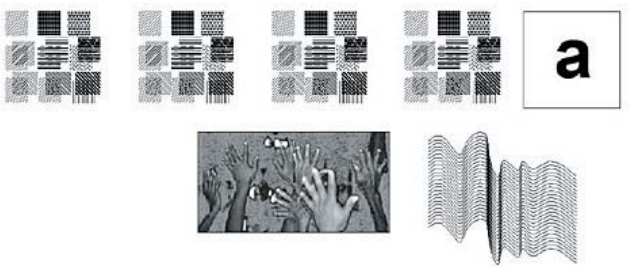
## Material

The following materials were analyzed:

Cotton and wool textiles, crimpline, polyester and fiber glass, mats of cork, silicon, latex, solid rubber, cellular rubber, foam rubber, acrylic sheets (transparent, translucent, colored), polypropylene, macrolon, epoxy resin and polyester resin, Teflon and polyamide plastic.

## Laser tests

High-resolution patterns (up to 600 dpi) were engraved into the material surface using table laser "X-660 50W" by Universal Laser Systems. Most of the materials were flatsheets, material thickness didn't exceed 1cm. Engraved vector graphic patterns had dimensions of 1,5/1,5 cm to 6/6 cm and hatches of different direction and density (angle range from 30-80 degrees and density from 0.5mm-2mm). Pixel graphics (pictures and characters in jpeg, gif and bitmap format) had dimensions of 2/2 cm to 11/5,5 cm and were transformed into grayscale (laser cannot handle colors).




*Lasered patterns*

## Procedure

The lasercutter X-660 has a cutting area of 812 x 457mm.  Power, speed and resolution of the laser path can be controlled by the user.

Settings range from 1% to 100%. Low speed combined with high power allow cutting the material while low power and high speed enable engraving of patterns or pictures. The duration of the process depends on chosen settings and graphic mode (pixel graphics take up to 500% longer then vector graphics).

A series of vector and pixel graphics were engraved using different settings of power and speed into each material sample. The results are documented in table form in the annex of the NDS group booklet. They provide information about materiality, reference address, laser settings, reaction and appearance of the engraved material.


*Page from documentation*

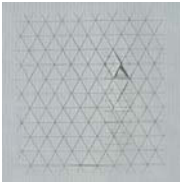
## Results

A wide scope of material performance was noticed during the testing process. Natural materials like cotton, wool and cork leave dark brown traces of burn-ups (carbon) and are less stable after the engraving (you can tear them easily). Synthetic materials like polyester show barely visible or yellowish-brown traces and melt while high power of laser (70-90%) or too low speed (5-20%) were used.

The precision of the engraving depends on the structure and texture of the material. Highly structured, woven materials with rough surfaces seem to leave less precise engravings than even surfaces.

Milled and engraved fonts

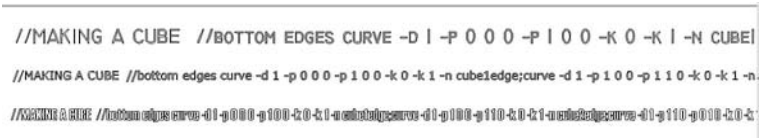
***Lasered cotton sample***



***Lasered polyester sample***

In the second part of the research fonts of different sizes were engraved using 3-axis CNC milling machine (Precix) into diverse surfaces. The following materials were chosen:

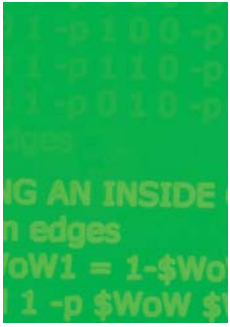
Acrylic panel (5 mm, transparent, dark blue)
Acrylic panel (3 mm, translucent, green)
HardCore-Panel (Trespa, 8 mm, 70% wood fibre, 30% synthetics)
Multilayered plywood panel (22 mm, dark brown, coated).




*Milled fonts*

Unlike vector graphics pictures need a lot more time to be engraved. Therefore the fonts are engraved into the 5 mm acrylic panel as vector outlines with hatches. The result was encouraging, the denser the hatches are the better the text can be read, even from a distance.

*Acrylic panel*

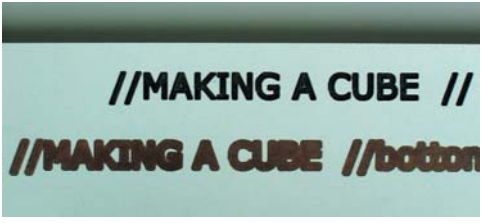


*Milled acrylic panel*

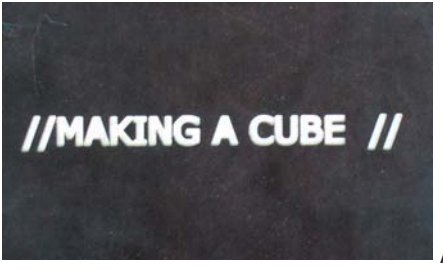
The 3 mm acrylic panel was engraved with a 3 mm flat nose bit. The font needed to be changed from true-type to polyline to be read properly as an outline by the milling machine. The diameter of the drill should not be bigger than the distance between the outlines otherwise some stock material was left. Curves of the outlines were milled very rough, edgy, the result was barely acceptable.

Trespa panel and multilayered plywood were engraved using font “Verdana” in 10 mm and 7 mm letter height.

The 2 mm bit was considered as too big, the fonts were very dense, some characters touched each other. A 1 mm bit gave a precise, clear and readable result.




*Milled Trespa panel*




*Milled multilayered plywood*

## Fixation to frames

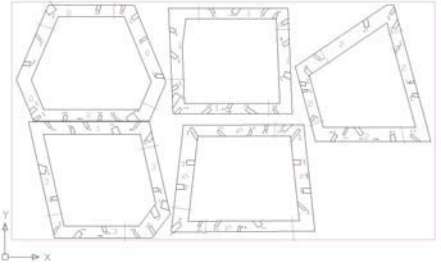
Two ways of fixture of the acrylic sheets to the pavilion frame were examined

-screwing the panel directly to the frame
-attaching the panel usin cable suspension. Both possibilities were not satisfying and were discarded.

*Panel screwed to the frame, panel suspended on wires*

## Lasered mock-ups

The last part of the research included making of various mock-ups of the pavilion in different sizes and out of different materials. The size of the pavilion, number and thickness of planes and thickness of the material are taken as parameters into Maya, using Mel Script. Then the file could be exported into AutoCAD, in dxf format. Files needed to be checked for missing slots in AutoCAD and then changed into a 2-dimensional drawing. Each frame of the drawing could now be labeled, lasered and put up according to the numbers.

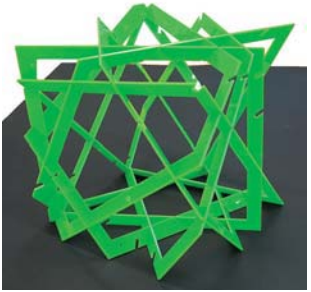


*CAD layout of frames*

1.Example of an mock-up made of 4 mm black cellular foam (Flexathen EZ 45), dimensions: 20/20/20 cm. The material is flexible and could be lasered within short time (20 min.) with settings 20/7 (power/ speed).

2.Example of a mock-up lasered in 2*4 mm corrugated cardboard, 40/40/40 cm in size.

3.Example of an mock-up built of green acrylic sheets, dimensions: 30/30/30cm, thickness:3mm.
Lasering time: 3 hours with power of 100% and speed of 1%.
The material is fragile, brakes easily while bent.



4.Example of an mock-up built of orange acrylic sheets, dimensions: 18/18/18cm,thickness:3mm. Lasering time: 1.5 hours with power of 100% and speed of 1%.
The material is very fragile, it brakes easily while bent.
Slots were too long, it was impossible to adjust parts without breaking them at some places.